EXHIBIT 8

EXHIBIT A

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

Chart Detailing Defendant's Infringement of U.S. Patent No. 8,924,192

Wapp Tech Ltd. & Wapp Tech Corp. v. J.P. Morgan Chase Bank, N.A.,
Case No. 4:23-cv-1137-ALM (E.D. Tex.)

The Accused Instrumentalities include tools from Google used to develop applications for Android mobile devices, including Android Studio, Android Emulator, Android Virtual Devices, Android Profiler, and Android App Inspection tools.

Based on the information presently available to them, Plaintiffs Wapp Tech Limited Partnership and Wapp Tech Corp. ("Wapp" or "Plaintiffs") are informed and believe that Defendant directly and indirectly infringes U.S. Patent No. 8,924,192 (the "'192 Patent"). Defendant directly infringes the '192 Patent claims when its employees, agents, and/or representatives use the Accused Instrumentalities to develop applications for mobile devices. Defendant directly infringes claims 60, 61, 62, and 65 of the '192 Patent when it makes, uses, and/or sells its applications for mobile devices. Upon information and belief, to the extent Defendant uses third parties in the development process, Defendant indirectly infringes the '192 Patent by actively inducing the direct infringement of third parties contracted to use the Accused Instrumentalities to develop applications for mobile devices on Defendant's behalf.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

Table of Contents

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[A] A system for developing an application for a mobile device comprising:

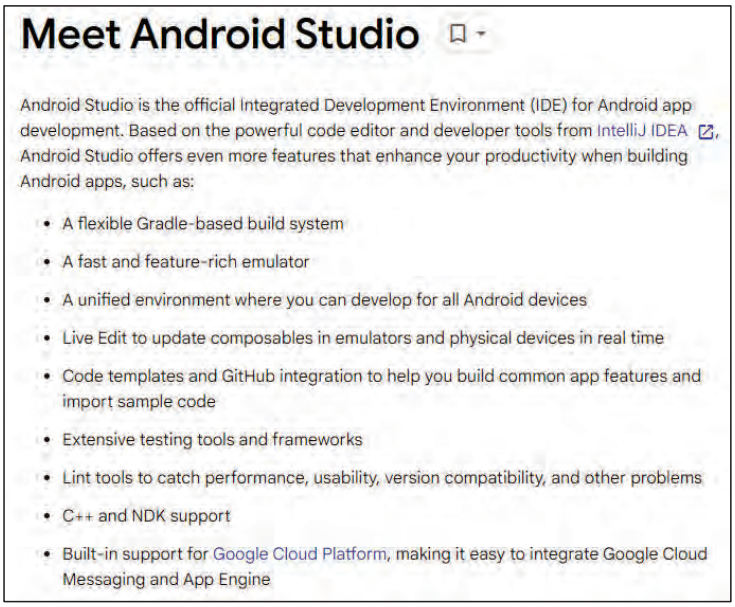
Claim 1

1[A] A system for developing an application for a mobile device comprising:

The Accused Instrumentalities are a system for developing an application for a mobile device. Defendant develops its mobile banking applications through its use of the Accused Instrumentalities by writing source code for the application, compiling that source code, and then testing the code. Android Studio is an Integrated Development Environment (IDE) for developing applications based on the Android operating system. The Android operating system runs on various mobile devices, including smartphones, tablets, and wearables. Android Studio includes "[a] fast and feature-rich emulator" and "[a] unified environment where you can develop for all Android devices."

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[A] A system for developing an application for a mobile device comprising:



Meet Android Studio

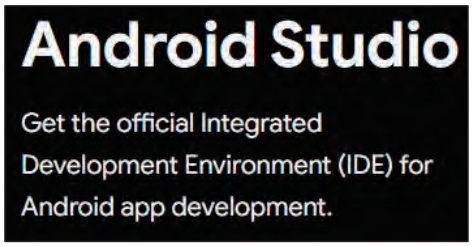
Android Studio is the official Integrated Development Environment (IDE) for Android app development. Based on the powerful code editor and developer tools from IntelliJ IDEA, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Live Edit to update composables in emulators and physical devices in real time
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

https://developer.android.com/studio/intro (last visited 4/6/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[A] A system for developing an application for a mobile device comprising:



https://developer.android.com/studio (last visited 4/6/2024).

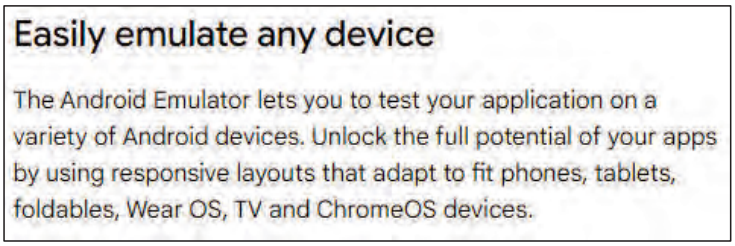
Android Studio includes a code editor for developing mobile device applications.

https://developer.android.com/studio (last visited 4/6/2024).

Android Studio includes an emulator for developing mobile device applications. The emulator allows application authors to "test [their] application on a variety of Android devices."

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[A] A system for developing an application for a mobile device comprising:



https://developer.android.com/studio (last visited 4/6/2024).

On information and belief, Defendant uses Android Studio to develop mobile banking applications for its business, for example—Chase Mobile. While Chase Mobile is identified as an example application, the contentions detailed in this chart apply to all mobile application development done by or on behalf of Defendant using Android Studio. On information and belief, Defendant's development of mobile banking applications includes using the features detailed throughout this document.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

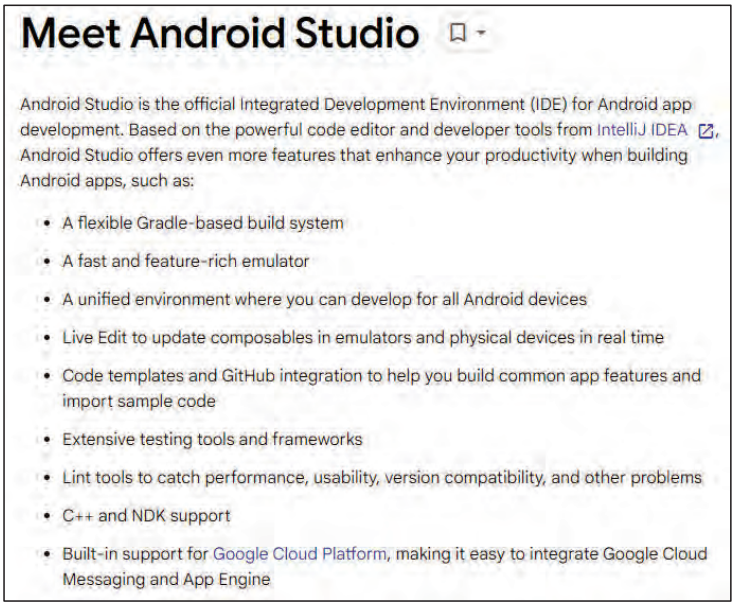
The Accused Instrumentalities include a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application.

1. **"software authoring interface"**

Android Studio includes a software authoring interface. Android Studio is an Integrated Development Environment (IDE) for developing applications based on the Android operating system. Developers use Android Studio to develop applications by writing source code and compiling that source code into programs that will run on Android devices. Android Studio is built on top of a "powerful code editor and developer tools," and it includes "[a] fast and feature-rich emulator," and "[a] unified environment where you can develop for all Android devices."

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



Meet Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development. Based on the powerful code editor and developer tools from IntelliJ IDEA, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

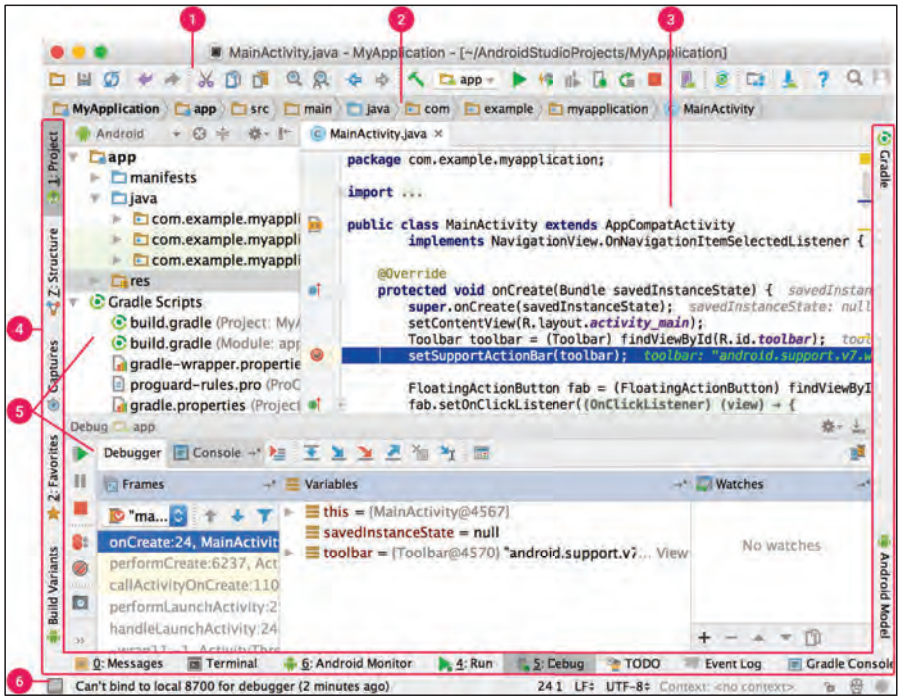
- A flexible Gradle-based build system
- A fast and feature-rich emulator
- A unified environment where you can develop for all Android devices
- Live Edit to update composables in emulators and physical devices in real time
- Code templates and GitHub integration to help you build common app features and import sample code
- Extensive testing tools and frameworks
- Lint tools to catch performance, usability, version compatibility, and other problems
- C++ and NDK support
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

https://developer.android.com/studio/intro (last visited 4/6/2024).

The following figure illustrates the Android Studio main window that is used for viewing and editing source code files.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

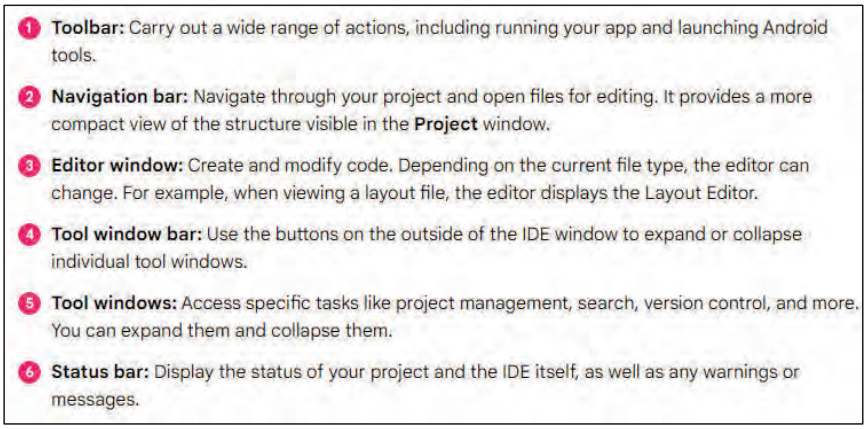
1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



https://developer.android.com/studio/intro/user-interface (last visited 4/6/2024). The editor window (#3) is used to create and modify source code, which is a form of software authoring. The remaining windows are described below:

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



https://developer.android.com/studio/intro/user-interface (last visited 4/6/2024). Android Studio's software authoring interface includes a number of different "windows" or tools that are available to the application author throughout the authoring process. Such tools and windows within Android Studio are part of its software authoring interface.

2. "configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application"

The Accused Instrumentalities are configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application.

a. Simultaneous visual emulation of network characteristics

Android Studio supports simultaneous visual emulation of network characteristics through the use of the Android Emulator and Android Virtual Devices (AVDs). The Android Emulator can be used to emulate/simulate Android devices on a computer without actually possessing the physical device. The Android Emulator emulates/simulates "almost all the capabilities of a real Android device."

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

Android Emulator includes predefined device configurations (or hardware profiles), and typically "the emulator is the best option for your testing needs."

> The Android Emulator simulates Android devices on your computer so that you can test your application on a variety of devices and Android API levels without needing to have each physical device. The emulator offers these advantages:
>
> - **Flexibility**: In addition to being able to simulate a variety of devices and Android API levels, the emulator comes with predefined configurations for various Android phone, tablet, Wear OS, and Android TV devices.
> - **High fidelity**: The emulator provides almost all the capabilities of a real Android device. You can simulate incoming phone calls and text messages, specify the location of the device, simulate different network speeds, simulate rotation and other hardware sensors, access the Google Play Store, and much more.
> - **Speed**: Testing your app on the emulator is in some ways faster and easier than doing so on a physical device. For example, you can transfer data faster to the emulator than to a device connected over USB.
>
> In most cases, the emulator is the best option for your testing needs. This page covers the core emulator functionalities and how to get started with it.

https://developer.android.com/studio/run/emulator (last visited 4/6/2024).

The Android Emulator is available directly within Android Studio and runs inside Android Studio by default. However, the emulator can also be launched in a separate tool window from within Android Studio as well.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

> The Android Emulator runs inside Android Studio by default. This lets you use screen space efficiently, navigate quickly between the emulator and the editor window using hotkeys, and organize your IDE and emulator workflow in a single application window.
>
> However, some emulator features are only available when you run it in a separate window. To launch the emulator in a separate window, go to **File > Settings > Tools > Emulator** (**Android Studio > Preferences > Tools > Emulator** on macOS) and deselect **Launch in a tool window**.

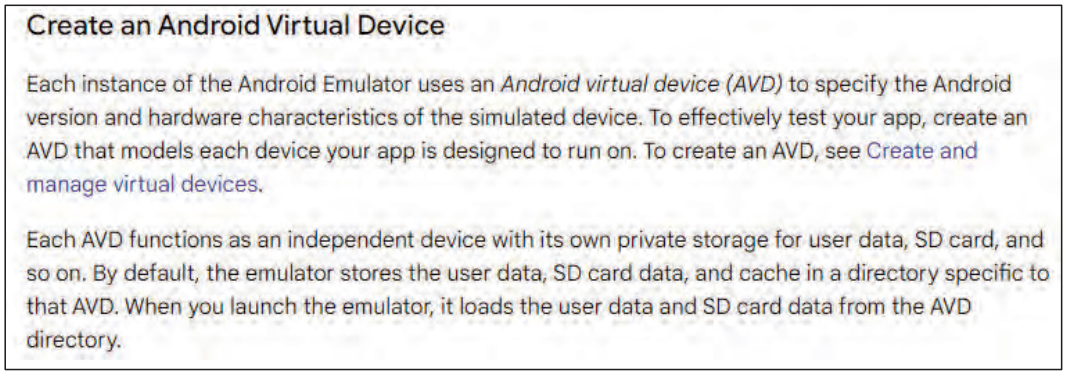
https://developer.android.com/studio/run/emulator-launch-separate-window (last visited 4/6/2024).

The device being emulated/simulated by Android Emulator is based on an Android Virtual Device (AVD). An AVD specifies the "hardware characteristics" of the device being emulated/simulated. Each AVD specifies the resources of the emulated/simulated device. These AVDs are created and managed in Android Studio using the Android Device Manager.[1]

[1] Prior to the Bumblebee release, Device Manager was referred to as the Android Virtual Device Manager.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



Create an Android Virtual Device

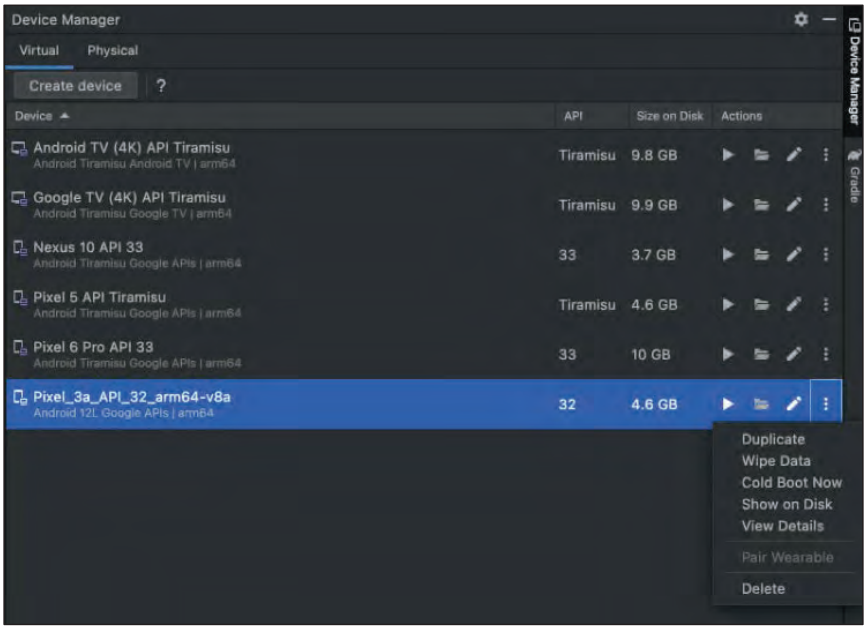
Each instance of the Android Emulator uses an *Android virtual device (AVD)* to specify the Android version and hardware characteristics of the simulated device. To effectively test your app, create an AVD that models each device your app is designed to run on. To create an AVD, see Create and manage virtual devices.

Each AVD functions as an independent device with its own private storage for user data, SD card, and so on. By default, the emulator stores the user data, SD card data, and cache in a directory specific to that AVD. When you launch the emulator, it loads the user data and SD card data from the AVD directory.

https://developer.android.com/studio/run/emulator (last visited 4/6/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

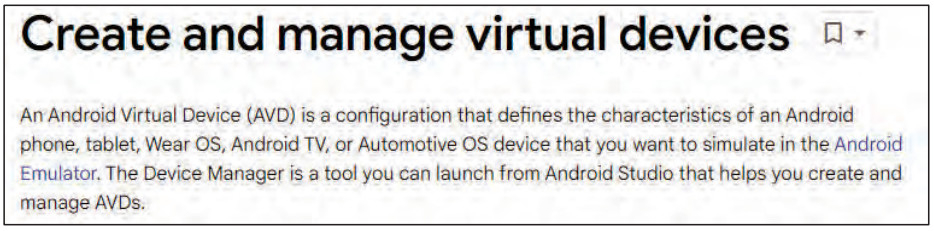


https://developer.android.com/studio/run/managing-avds (last visited 4/6/2024) (illustrating Android Device Manager).

Once an Android Virtual Device is created, its operating properties can be specified. All of these properties are simulated/ emulated during operation within the Emulator.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



https://developer.android.com/studio/run/managing-avds (last visited 4/6/2024).

Each Android Virtual Device has a profile that includes a number of properties defining the characteristics of the AVD to emulate/simulate. The "hardware profile properties" include:

- Device Name
- Device Type
- Screen: Screen Size
- Screen: Screen Resolution
- Screen: Round
- Memory: RAM
- Input: Has Hardware Buttons (Back/Home/Menu)
- Input: Has Hardware Keyboard
- Input: Navigation Style
- Supported Device States
- Cameras
- Sensors: Accelerometer
- Sensors: Gyroscope
- Sensors: GPS
- Sensors: Proximity Sensor
- Default Skin.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

See https://developer.android.com/studio/run/managing-avds (last visited 4/6/2024) (listing and describing hardware profile properties). Additionally, an AVD has AVD properties that also control the manner in which the AVD performs during emulation/simulation. The AVD Properties include:

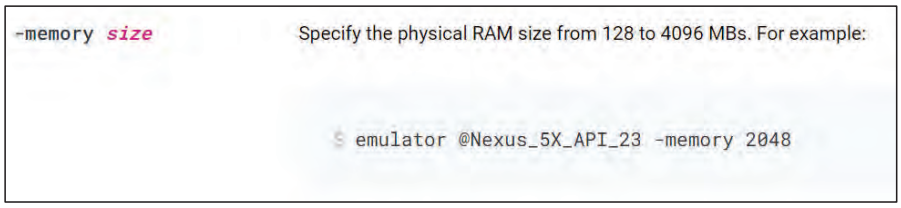
- AVD Name
- AVD ID (Advanced)
- Hardware Profile
- System Image
- Startup Orientation
- Camera (Advanced)
- Network: Speed (Advanced)
- Network: Latency (Advanced)
- Emulated Performance: Graphics
- Emulated Performance: Boot option (Advanced)
- Emulated Performance: Multi-Core CPU (Advanced)
- Memory and Storage: RAM (Advanced)
- Memory and Storage: VM Heap (Advanced)
- Memory and Storage: Internal Storage (Advanced)
- Memory and Storage: SD Card (Advanced)
- Device Frame: Enable Device Frame
- Custom Skin Definition (Advanced)
- Keyboard: Enable Keyboard Input (Advanced)

See https://developer.android.com/studio/run/managing-avds (last visited 4/6/2024) (listing and describing AVD properties). Each of the above-mentioned properties are emulated/simulated by Android Emulator when running the device profile specified for that particular AVD. The hardware profile properties and the AVD properties represent device characteristics, including hardware characteristics and network characteristics. When the Emulator runs a particular AVD, these characteristics are emulated/simulated and are indicative of the performance of the emulated/simulated device when running the application.

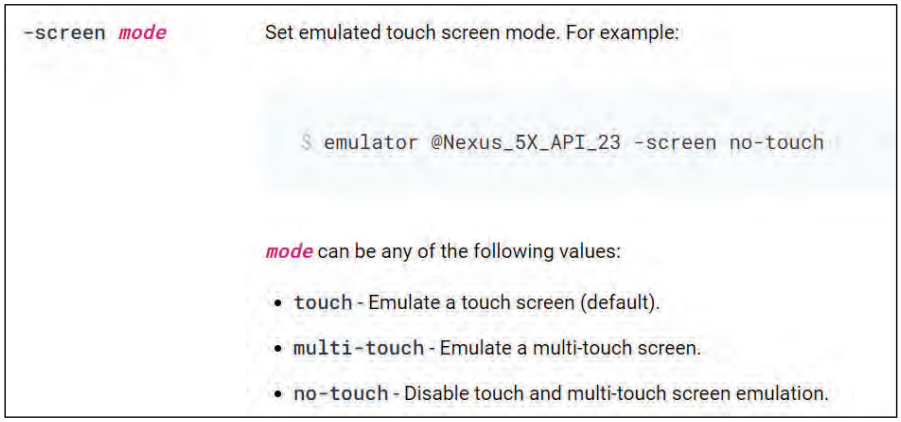
Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

Some hardware profile properties and AVD properties can also be controlled via the command-line options available for the Android Emulator. Examples of the command-line options permitting control of hardware characteristics of the AVD device during simulation are provided below.



https://developer.android.com/studio/run/emulator-commandline (last visited 4/6/2024) (showing options for controlling the RAM size of the emulated/simulated device).



https://developer.android.com/studio/run/emulator-commandline (last visited 4/6/2024) (showing options for controlling the touch capabilities to emulate/simulate for the device screen).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

The Android Emulator is programmed to emulate a plurality of network characteristics, such as network speed (upload/download) and network latency.

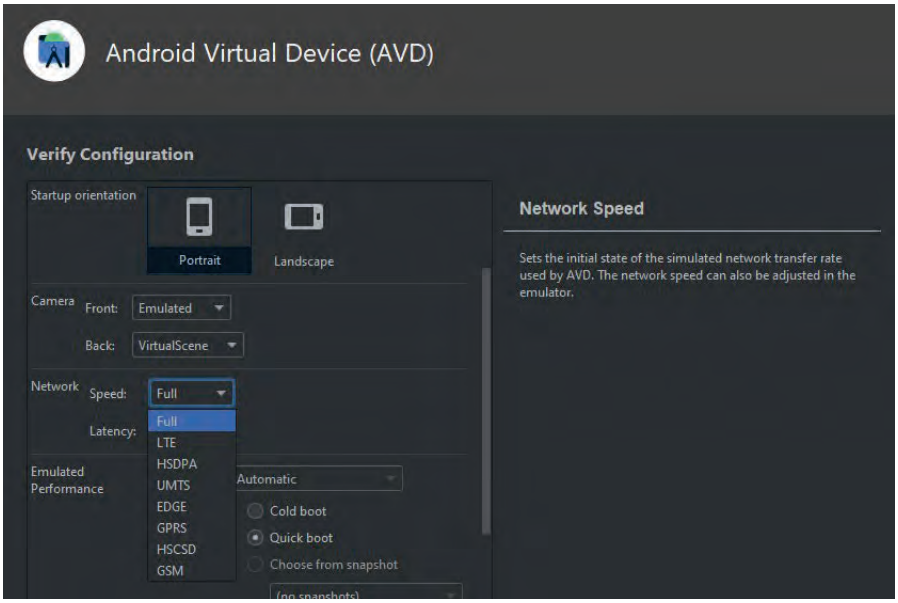
> The emulator supports network throttling as well as higher connection latencies. You can define it either through the skin configuration or with the `-netspeed` and `-netdelay` options.

https://developer.android.com/studio/run/emulator-commandline (last visited 4/6/2024).

The following figures show screenshots of the configuration screens for an Android Virtual Device. The first figure shows how an AVD can have a specific network speed associated with it, the speed options including: Full, LTE, HSDPA, UMTS, EDGE, GPRS, HSCSD, GSM.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



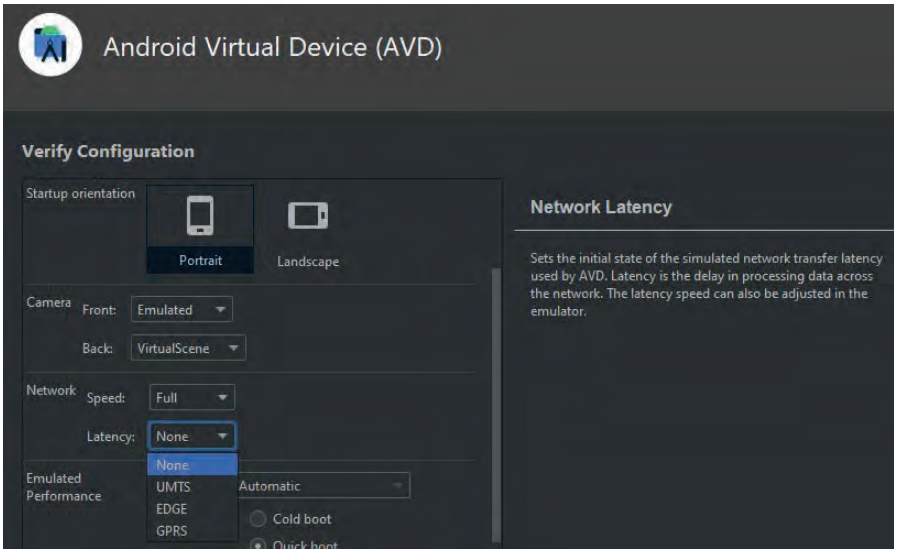
Screenshot from Android Studio Arctic Fox[2]: Android Virtual Device (showing Network Speed options: Full, LTE, HSDPA, UMTS, EDGE, GPRS, HSCSD, GSM).

The following screenshot shows how an AVD can have a specific network latency associated with it, the latency options including, for example: None, UMTS, EDGE, GPRS.

[2] New versions of Android Studio are released on a regular basis. The user interface may differ slightly between different versions; however, the functionality identified in this chart exists in Android Studio versions from 2017 to the present. To the extent the functionality in different versions of Android Studio changes in a meaningful way regarding the infringement read, those changes will be noted in the chart.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

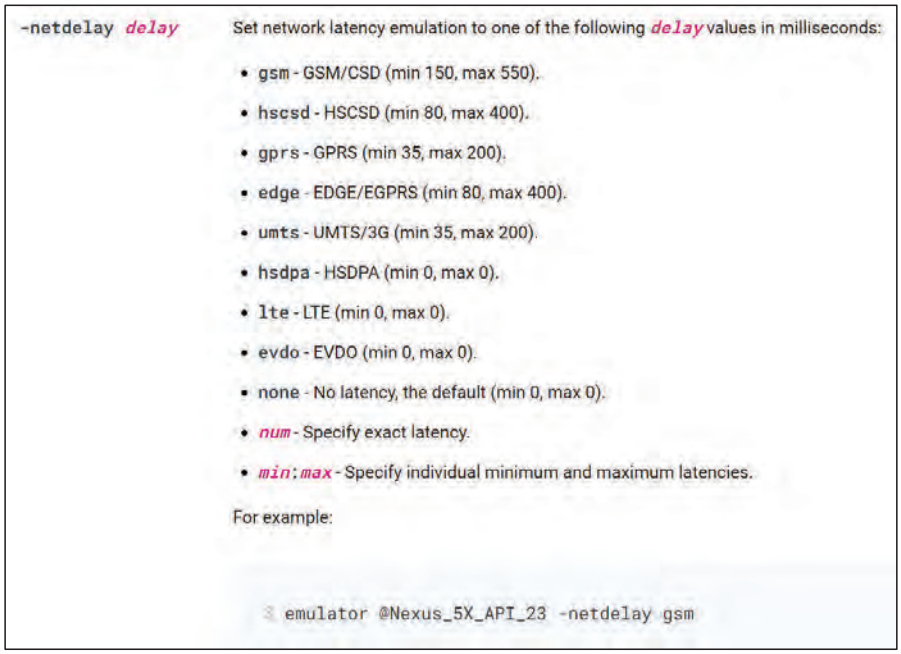


Screenshot from Android Studio Arctic Fox: Android Virtual Device (showing Network Latency options).

The network characteristics can also be controlled via the command-line options available for the Android Emulator, as illustrated below.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



`-netdelay` *delay*

Set network latency emulation to one of the following *delay* values in milliseconds:

- `gsm` - GSM/CSD (min 150, max 550).
- `hscsd` - HSCSD (min 80, max 400).
- `gprs` - GPRS (min 35, max 200).
- `edge` - EDGE/EGPRS (min 80, max 400).
- `umts` - UMTS/3G (min 35, max 200).
- `hsdpa` - HSDPA (min 0, max 0).
- `lte` - LTE (min 0, max 0).
- `evdo` - EVDO (min 0, max 0).
- `none` - No latency, the default (min 0, max 0).
- *num* - Specify exact latency.
- *min*`:`*max* - Specify individual minimum and maximum latencies.

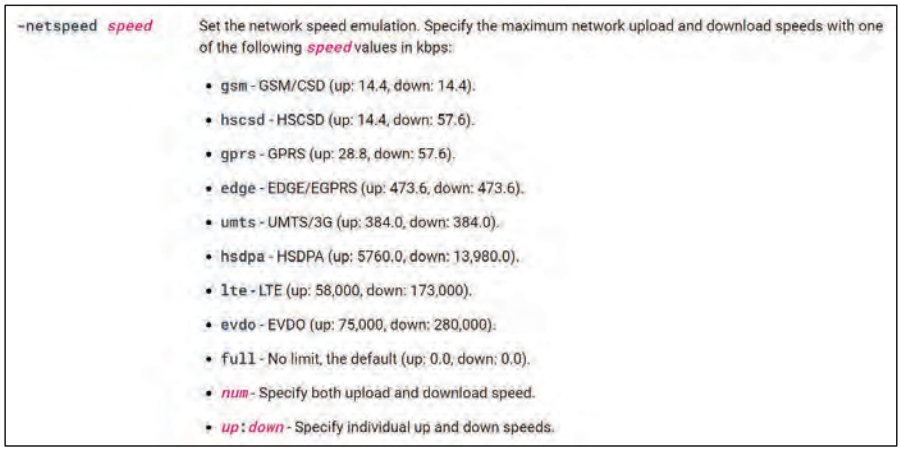
For example:

```
emulator @Nexus_5X_API_23 -netdelay gsm
```

https://developer.android.com/studio/run/emulator-commandline (last visited 4/7/2024) (showing options for setting network latency properties).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



`-netspeed` *speed*	Set the network speed emulation. Specify the maximum network upload and download speeds with one of the following *speed* values in kbps: • `gsm` - GSM/CSD (up: 14.4, down: 14.4). • `hscsd` - HSCSD (up: 14.4, down: 57.6). • `gprs` - GPRS (up: 28.8, down: 57.6). • `edge` - EDGE/EGPRS (up: 473.6, down: 473.6). • `umts` - UMTS/3G (up: 384.0, down: 384.0). • `hsdpa` - HSDPA (up: 5760.0, down: 13,980.0). • `lte` - LTE (up: 58,000, down: 173,000). • `evdo` - EVDO (up: 75,000, down: 280,000). • `full` - No limit, the default (up: 0.0, down: 0.0). • *num* - Specify both upload and download speed. • *up*`:`*down* - Specify individual up and down speeds.

https://developer.android.com/studio/run/emulator-commandline (last visited 4/7/2024) (showing options for setting network speed properties).

These network characteristics (e.g., speed and latency) are indicative of the performance of a mobile device, affecting the speed and latency with which it can communicate over a network connection while executing the application.

b. Profile display windows

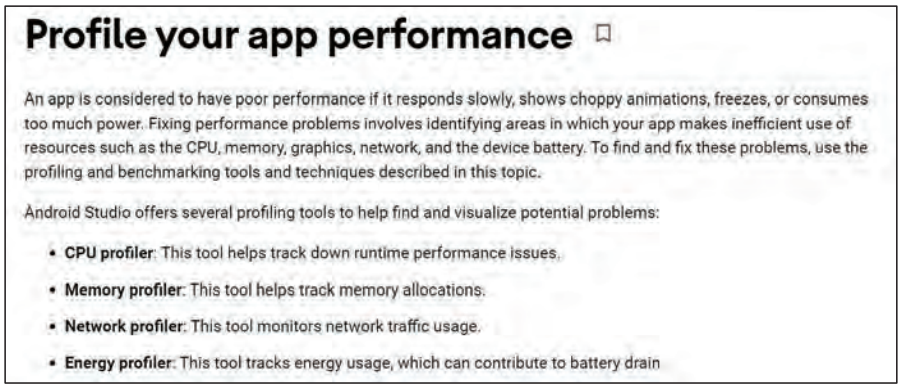
Android Studio includes a number of profiling tools that support application development, allowing a software author to monitor the resources of the Android device or AVD that are used by and available to the application while executing on the device. These profiling tools have corresponding display windows.

The Android profiling tools for Arctic Fox and relevant prior releases include: (1) CPU Profiler; (2) Memory Profiler; (3) Network Profiler; and (4) Energy Profiler. Starting with the Bumblebee release, the Network Profiler was moved to App Inspection and renamed the Network Inspector, as detailed further below.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

These profiling tools are used for "[f]ixing performance problems [which] involves identifying areas in which your app makes inefficient use of resources such as the CPU, memory, graphics, network, and the device battery . . . . Android studio offers several profiling tools to help find and visualize potential problems."



Profile your app performance

An app is considered to have poor performance if it responds slowly, shows choppy animations, freezes, or consumes too much power. Fixing performance problems involves identifying areas in which your app makes inefficient use of resources such as the CPU, memory, graphics, network, and the device battery. To find and fix these problems, use the profiling and benchmarking tools and techniques described in this topic.

Android Studio offers several profiling tools to help find and visualize potential problems:

- **CPU profiler**: This tool helps track down runtime performance issues.
- **Memory profiler**: This tool helps track memory allocations.
- **Network profiler**: This tool monitors network traffic usage.
- **Energy profiler**: This tool tracks energy usage, which can contribute to battery drain

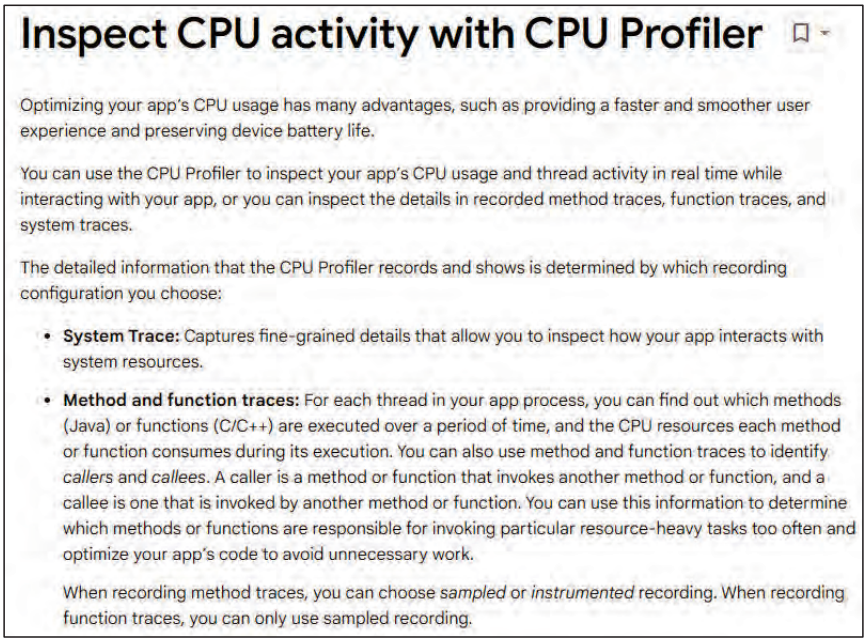
https://developer.android.com/studio/profile (last visited 1/6/2022).

Android Studio allows the application author to profile the CPU, memory usage, network activity, and energy use of the mobile device while executing an application. Each profiler is detailed below.

The **CPU Profiler** is used to monitor CPU usage and availability, and it helps track down runtime performance issues. It can be used to "inspect your app's CPU usage and thread activity in real time while interacting with your app . . . ."

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



Inspect CPU activity with CPU Profiler

Optimizing your app's CPU usage has many advantages, such as providing a faster and smoother user experience and preserving device battery life.

You can use the CPU Profiler to inspect your app's CPU usage and thread activity in real time while interacting with your app, or you can inspect the details in recorded method traces, function traces, and system traces.

The detailed information that the CPU Profiler records and shows is determined by which recording configuration you choose:

- **System Trace:** Captures fine-grained details that allow you to inspect how your app interacts with system resources.
- **Method and function traces:** For each thread in your app process, you can find out which methods (Java) or functions (C/C++) are executed over a period of time, and the CPU resources each method or function consumes during its execution. You can also use method and function traces to identify *callers* and *callees*. A caller is a method or function that invokes another method or function, and a callee is one that is invoked by another method or function. You can use this information to determine which methods or functions are responsible for invoking particular resource-heavy tasks too often and optimize your app's code to avoid unnecessary work.

  When recording method traces, you can choose *sampled* or *instrumented* recording. When recording function traces, you can only use sampled recording.

https://developer.android.com/studio/profile/cpu-profiler (last visited 4/7/2024) (detailing the CPU Profiler). The CPU Profiler displays the executed application's CPU usage, CPU availability, and thread activity via timelines, as illustrated below.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

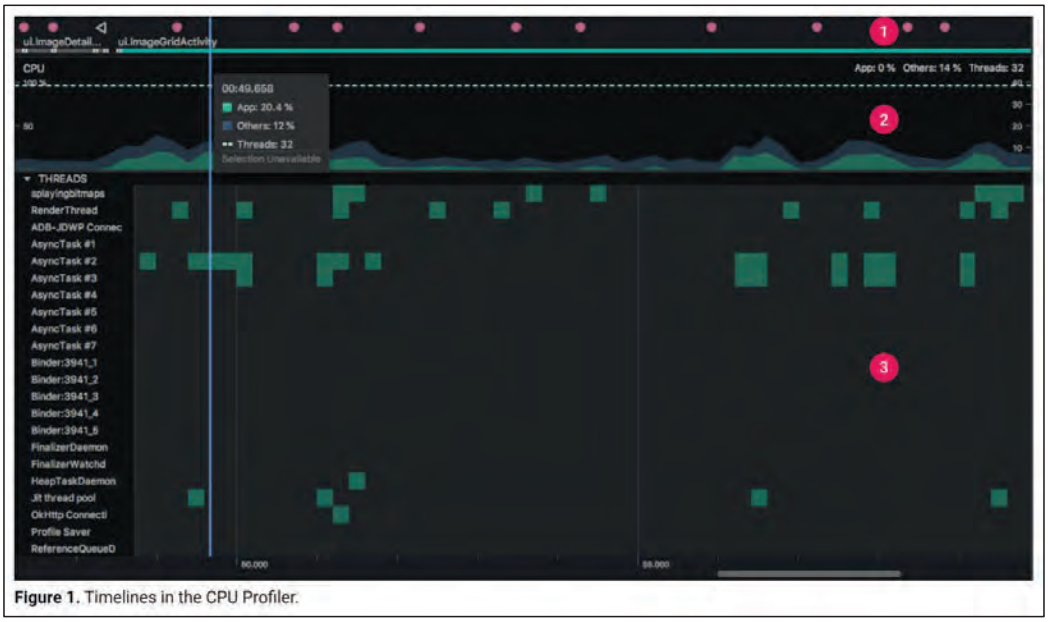


Figure 1. Timelines in the CPU Profiler.

https://developer.android.com/studio/profile/cpu-profiler (last visited 4/7/2024) (illustrating the CPU Profiler timelines).

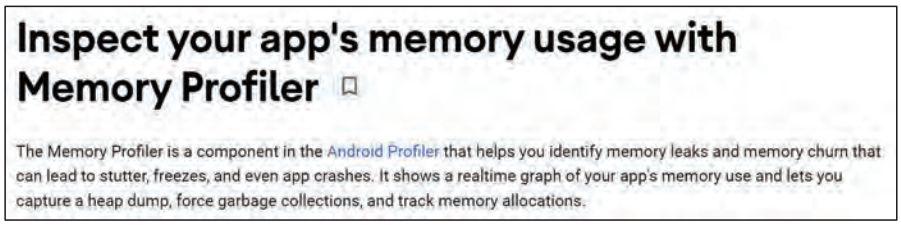
Referring to the numbers in the figure above, Number 1 illustrates the Event Timeline, which "[s]hows the activities in your app as they transition through different states in their lifecycle, and indicates user interactions with the device, including screen rotation events." *Id.* Number 2 illustrates the CPU Timeline, which "[s]hows real-time CPU usage of your app—as a percentage of total available CPU time—and the total number of threads your app is using. The timeline also shows the CPU usage of other processes (such as system processes or other apps), so you can compare it to your app's usage. You can inspect historical CPU usage data by moving your mouse along the horizontal axis of the timeline." *Id.* Finally, Number 3 indicates the Thread Activity Timeline, which "[l]ists each thread that belongs to your app process and indicates its activity along a timeline using the colors listed below." *Id.*

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

The CPU Timeline (Number 2) shows the CPU usage of the application and CPU availability during execution. In particular, the lighter green portions of the CPU timeline illustrate the percentage of CPU resources used by the application at any given point in time. The darker green/gray portion of the timeline shows the CPU resources consumed by other components on the device (e.g., system processes or other applications). Finally, the remaining dark/black portion of the timeline shows the amount of CPU resources available to the application. The dotted white line across the top of the CPU Timeline shows the 100% mark, indicating the maximum CPU resources available for consumption.

The **Memory Profiler** is used to monitor memory usage by the application as well as memory available to the application. It assists with detecting unwanted or unnecessary memory consumption, including memory leaks and memory churn.



https://developer.android.com/studio/profile/memory-profiler (last visited 2/7/2024) (describing the Memory Profiler). An example view of the Memory Profiler is provided below:

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

Figure 1. The Memory Profiler

https://developer.android.com/studio/profile/memory-profiler (last visited 4/7/2024) (illustrating the Memory Profiler). The memory legend near the top illustrates the amount of memory consumed or utilized by the application (e.g., Total, Java, Native, Graphics, Stack, Code, Others). Based on the AVD memory limit, this also indicates the total memory resources (unallocated and/or allocated) available to the application (i.e., hardware profile memory available minus total consumed). In addition, the Allocated component (represented in the graph by a white dotted line) indicates the amount of allocated memory resources currently available to the application. Further, the AVD includes a limit on the amount of memory (unallocated and/or allocated) available to the application while being simulated/emulated on the AVD device. In this way, the Memory Profiler provides information about the unallocated and allocated memory resources available to and utilized by an application for a given AVD configuration.

The **Network Profiler** allows the application author to monitor network connections and data exchanges by the mobile application.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

Inspect network traffic with Network Profiler

The Network Profiler displays realtime network activity on a timeline, showing data sent and received, as well as the current number of connections. This lets you examine how and when your app transfers data, and optimize the underlying code appropriately.

https://developer.android.com/studio/profile/network-profiler (last visited 1/6/2022).

As illustrated below, the Network Profiler shows the receiving network speed, the sending network speed, and latency.

https://developer.android.com/studio/profile/network-profiler (last visited 7/27/2021).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

The Connection View provides additional information about the network transmissions, including transmission duration and timing, which is related to network latency.

> • **Connection View**: Lists files that were sent or received during the selected portion of the timeline across all of your app's CPU threads. For each request, you can inspect the size, type, status, and transmission duration. You can sort this list by clicking any of the column headers. You also see a detailed breakdown of the selected portion of the timeline, showing when each file was sent or received.

https://developer.android.com/studio/profile/network-profiler (last visited 7/27/2021).

The Thread View displays the network activity for each of the application's CPU threads, illustrating the receiving network speed, the sending network speed, and latency.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

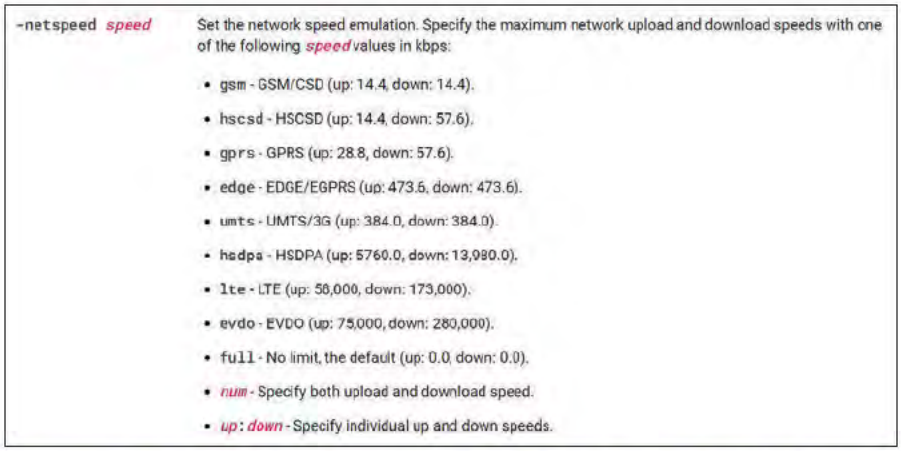
Figure 2. Inspecting network requests by app thread in the **Thread View**

https://developer.android.com/studio/profile/network-profiler (last visited 1/7/2022). The blue line in the network timeline shows the receiving rate of data flow, and the orange line shows the sending rate of data flow for the application. Each of these translate to bandwidth resources used by the application. The dotted line indicates the number of connections and the maximum data transfer rate employed by the application. The indication of a previously seen maximum data rate provides information about the bandwidth resources available to the application.

Additionally, as detailed above, the AVD is configured with network throttling properties that limit the bandwidth—both upload and download speeds—that is available to the application.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



`-netspeed` *speed*	Set the network speed emulation. Specify the maximum network upload and download speeds with one of the following *speed* values in kbps:

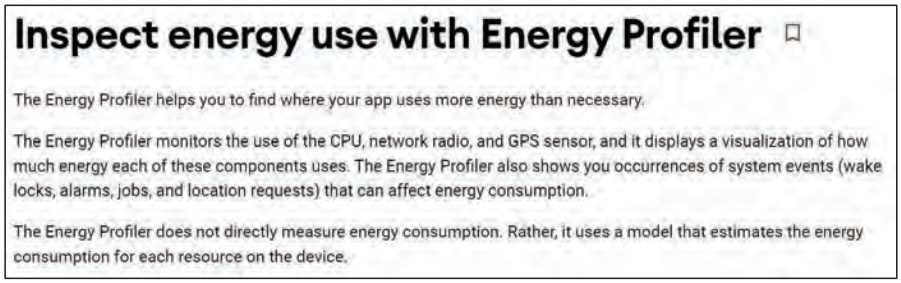
- `gsm` - GSM/CSD (up: 14.4, down: 14.4).
- `hscsd` - HSCSD (up: 14.4, down: 57.6).
- `gprs` - GPRS (up: 28.8, down: 57.6).
- `edge` - EDGE/EGPRS (up: 473.6, down: 473.6).
- `umts` - UMTS/3G (up: 384.0, down: 384.0).
- `hsdpa` - HSDPA (up: 5760.0, down: 13,980.0).
- `lte` - LTE (up: 58,000, down: 173,000).
- `evdo` - EVDO (up: 75,000, down: 280,000).
- `full` - No limit, the default (up: 0.0, down: 0.0).
- *num* - Specify both upload and download speed.
- *up*:*down* - Specify individual up and down speeds.

https://developer.android.com/studio/run/emulator-commandline (last visited 4/6/2024) (showing the upload and download maximums for different "network speed" settings provided in the AVD configuration). For example, an AVD configured with EDGE network speed is limited to 473.6 kbps upload speed and 473.6 kbps download speed. This provides an upper-bound on the network bandwidth resources available to the application, and the Network Profiler displays the utilized bandwidth per unit time, thus showing the remaining bandwidth available to the application based on its network speed settings.

The **Energy Profiler** allows the author to monitor energy consumption by the application.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



Inspect energy use with Energy Profiler

The Energy Profiler helps you to find where your app uses more energy than necessary.

The Energy Profiler monitors the use of the CPU, network radio, and GPS sensor, and it displays a visualization of how much energy each of these components uses. The Energy Profiler also shows you occurrences of system events (wake locks, alarms, jobs, and location requests) that can affect energy consumption.

The Energy Profiler does not directly measure energy consumption. Rather, it uses a model that estimates the energy consumption for each resource on the device.

https://developer.android.com/studio/profile/energy-profiler (last visited 4/7/2024). An example view of the Energy Profiler is provided below:

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

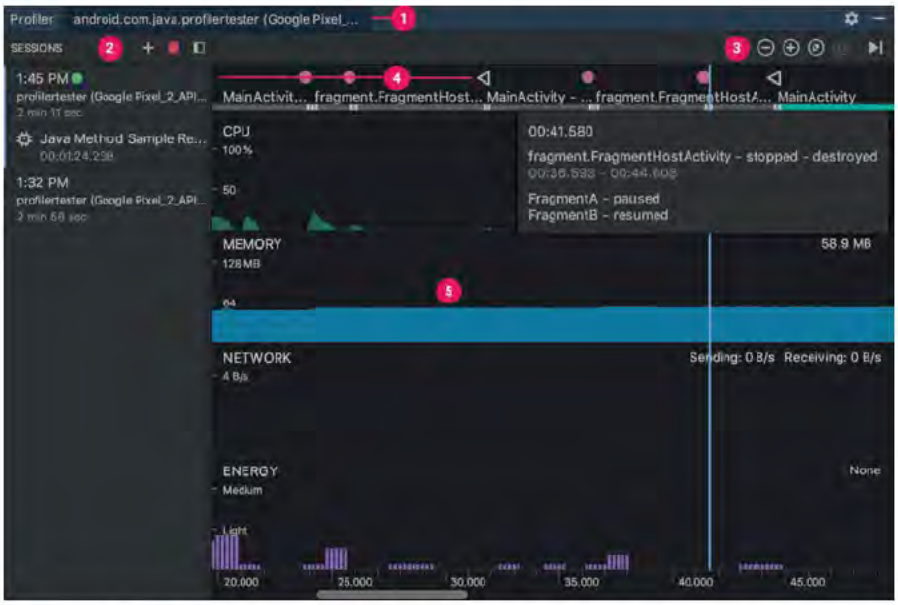
Figure 1. Timelines in the Energy Profiler.

https://developer.android.com/studio/profile/energy-profiler (last visited 4/7/2024). Three timelines are illustrated in this Energy Profiler view. The first (1) is the Event Timeline, which "[s]hows the activities in your app as they transition through different states in their lifecycle. This timeline also indicates user interactions with the device, including screen rotation events." *Id.* The second (2) is the Energy Timeline, which "[s]hows estimated energy consumption of your app." *Id.* And the third (3) is the System Time, which "[i]ndicates system events that may affect energy consumption." *Id.* By moving your mouse over the timelines, you can "see a breakdown of energy use by CPU, network, and location (GPS) resources . . . ." *Id.*

In addition to the exemplary profile display windows illustrated above, Android Studio Arctic Fox (and earlier versions) supports display of profile windows for all four profilers discussed above:

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;

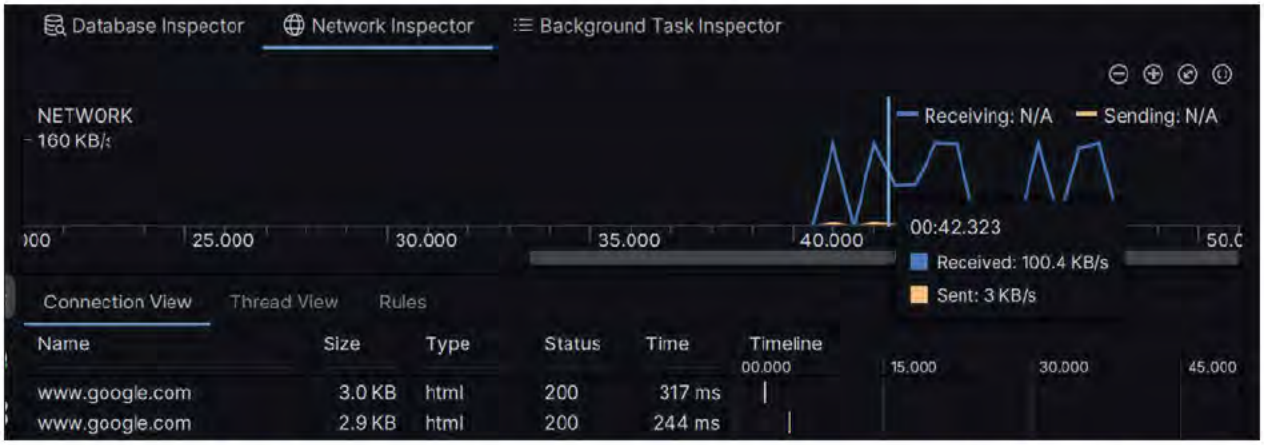


https://developer.android.com/studio/profile/android-profiler (last visited 7/27/2021) (illustrating the CPU, Memory, Network, and Energy profile displays).

Starting with the Android Studio Bumblebee release, the Network Profiler was moved to the App Inspection component of Android Studio and renamed Network Inspector. *See* https://developer.android.com/studio/releases/past-releases/as-bumblebee-release-notes (last visited 4/20/2024). The Network Inspector view is shown below:

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[B] a software authoring interface configured to simultaneously visually emulate, via one or more profile display windows, a plurality of network characteristics indicative of performance of the mobile device when executing the application;



Screenshot from Android Studio Iguana (showing Network Inspector). The Network Inspector view shows the receiving network speed and the sending network speed. And it includes the Connection View and Thread View discussed above for the Network Profiler.

The CPU and Memory Profilers are available in post-Arctic Fox versions of Android Studio, as detailed above for Arctic Fox.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

1[C] wherein the software authoring interface is further configured to simulate a network connection state encountered by the mobile device.

1[C] wherein the software authoring interface is further configured to simulate a network connection state encountered by the mobile device.

Android Studio is configured to simulate a network connection state encountered by the mobile device.

See 1[B] (detailing how the network speed and latency are emulated/simulated by selecting a network type, which indicates a connection state).

The selection of a well-known network type (*e.g.*, LTE, GSM) causes the Simulator to simulate/emulate that network type, which includes emulating/simulating the network connection states associated with such networks. Controlling through simulation the speed and latency of a network interacting with the mobile device is simulating a network connection state encountered by the mobile device. The network connection states include the options available for Network Speed and Network Latency (illustrated in the dropdowns above in 1[B]): Full, LTE, HSDPA, UMTS, EDGE, GPRS, HSCSD, GSM, None.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

2[A] The system of claim 1, wherein the software authoring interface is configured to enable a user to select from one or more connection simulations for testing how well mobile content performs on the mobile device.

Claim 2

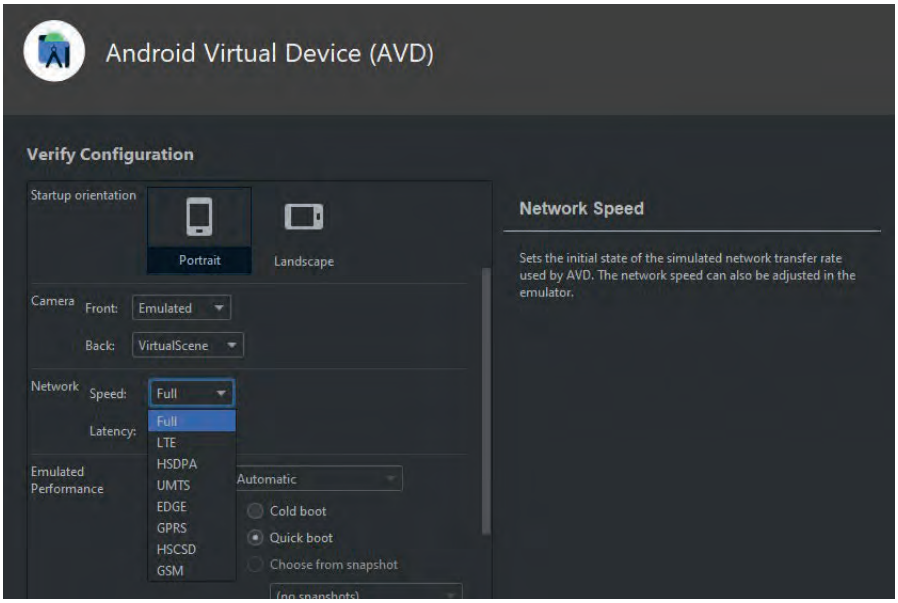
2[A] The system of claim 1, wherein the software authoring interface is configured to enable a user to select from one or more connection simulations for testing how well mobile content performs on the mobile device.

Android Studio is configured to enable a user to select from one or more connection simulations for testing how well mobile content performs on the mobile device.

The figure below illustrates how Android Studio permits a user to select the type of the network connection, which will affect the bandwidth available to the mobile device and thus how well mobile content performs on the mobile device.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

2[A] The system of claim 1, wherein the software authoring interface is configured to enable a user to select from one or more connection simulations for testing how well mobile content performs on the mobile device.

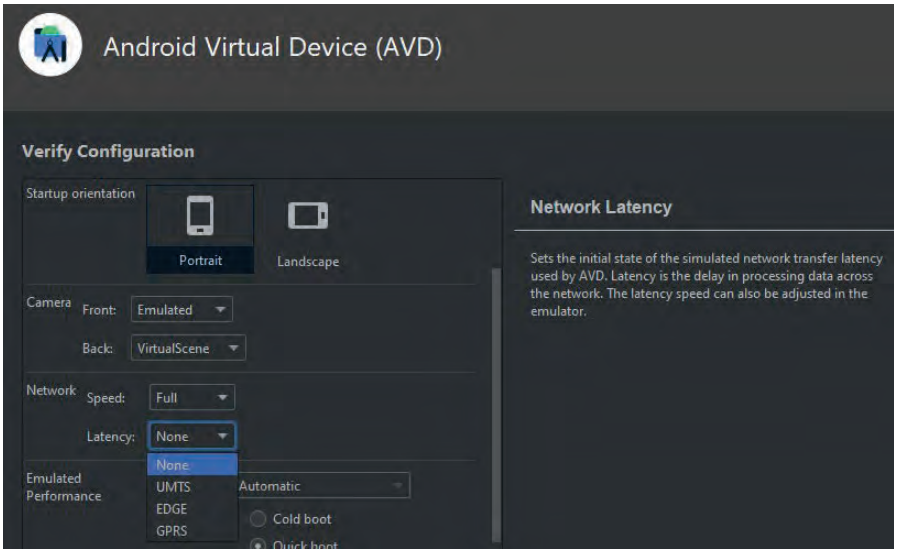


Screenshot from Android Studio Arctic Fox: Android Virtual Device (showing Network Speed options).

The figure below illustrates how Android Studio permits a user to select the latency of the network connection, which will affect the mobile device's performance on the network and thus how well mobile content performs on the mobile device.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

2[A] The system of claim 1, wherein the software authoring interface is configured to enable a user to select from one or more connection simulations for testing how well mobile content performs on the mobile device.



Screenshot from Android Studio Arctic Fox: Android Virtual Device (showing Network Latency options).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

3[A] The system of claim 2, wherein the one or more connection simulations are configured to simulate wireless transmission of content to the mobile device based on the selected connection simulation.

Claim 3

3[A] The system of claim 2, wherein the one or more connection simulations are configured to simulate wireless transmission of content to the mobile device based on the selected connection simulation.

Android Studio is configured to configured to simulate wireless transmission of content to the mobile device based on the selected connection simulation. For example, when a user selects a wireless network such as EDGE or LTE in the AVD Properties, then Android Emulator will simulate wireless transmission of content to the mobile device by simulating network characteristics that would occur when transmitting wireless content to the mobile device. *See* 2[A] (illustrating selection of network AVD Properties)

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

4[A] The system of claim 2, wherein the connection simulation includes one or more profiles.

Claim 4

4[A] The system of claim 2, wherein the connection simulation includes one or more profiles.

In Android Studio, each connection simulation (*e.g.*, EDGE, LTE, etc.) includes at least one profile dictating the corresponding speed and/or latency for that named connection simulation. *See* 2[A] (illustrating the connection simulations).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

5[A] The system of claim 4, wherein the profiles include preset profiles.

Claim 5

5[A] The system of claim 4, wherein the profiles include preset profiles.

In Android Studio, the profiles for each of the connection simulations detailed above regarding limitation 2[A] are preset profiles. Their corresponding speed and/or latency settings are preset.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

6[A]The system of claim 4, wherein the profiles are configured to enable a user to manage the profiles.

Claim 6

6[A] The system of claim 4, wherein the profiles are configured to enable a user to manage the profiles.

As detailed below, *see* 7[A], the profiles are configured to enable a user to create custom profiles. Creating custom profiles is a form of managing profiles.

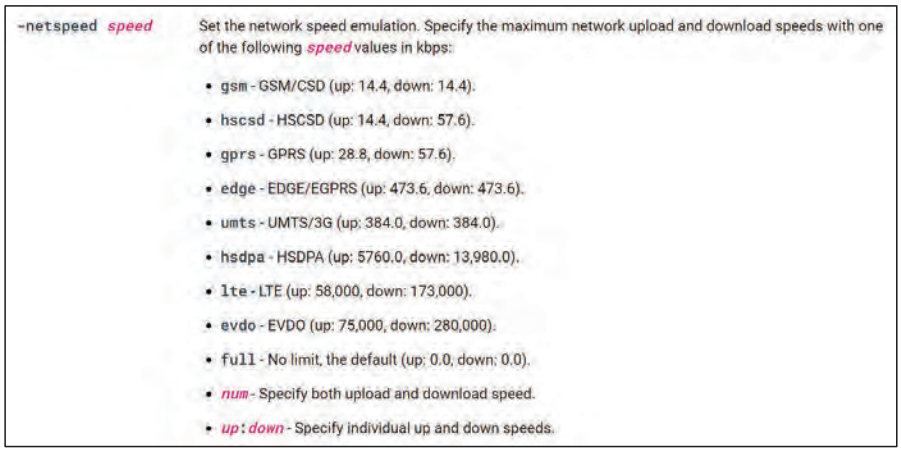
Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.

Claim 7

7[A] The system of claim 4, wherein the profiles are configured to enable a user to create custom profiles.

Android Studio is configured to allow a user to create custom profiles by running the emulator from the command-line with the -netdelay and/or -netspeed options. These options allow a user to specify a minimum and maximum network latency as well as an upload and download speed for the simulated network connections.

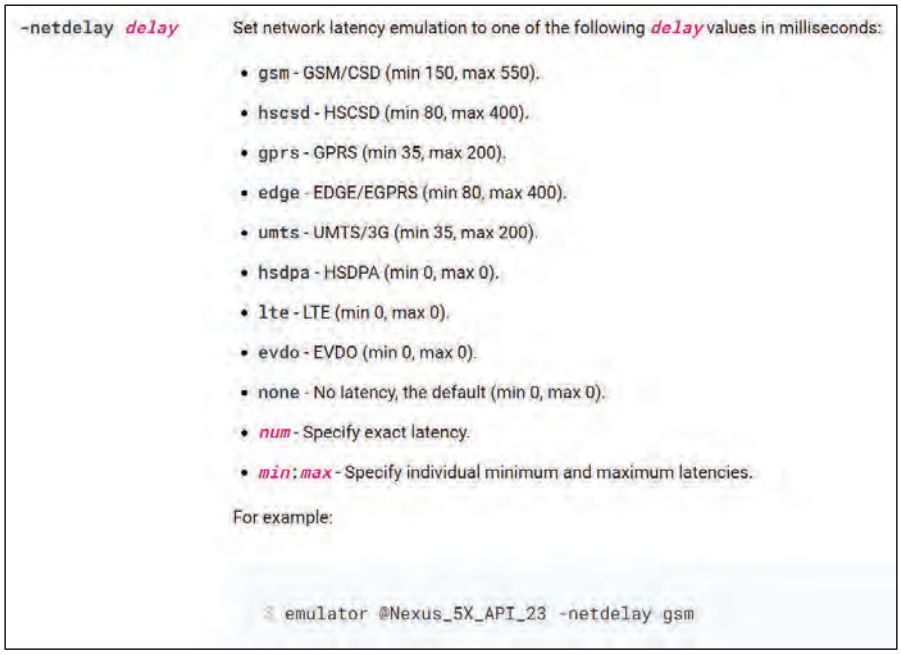


`-netspeed` *speed*	Set the network speed emulation. Specify the maximum network upload and download speeds with one of the following *speed* values in kbps: • `gsm` - GSM/CSD (up: 14.4, down: 14.4). • `hscsd` - HSCSD (up: 14.4, down: 57.6). • `gprs` - GPRS (up: 28.8, down: 57.6). • `edge` - EDGE/EGPRS (up: 473.6, down: 473.6). • `umts` - UMTS/3G (up: 384.0, down: 384.0). • `hsdpa` - HSDPA (up: 5760.0, down: 13,980.0). • `lte` - LTE (up: 58,000, down: 173,000). • `evdo` - EVDO (up: 75,000, down: 280,000). • `full` - No limit, the default (up: 0.0, down: 0.0). • *num* - Specify both upload and download speed. • *up*`:`*down* - Specify individual up and down speeds.

https://developer.android.com/studio/run/emulator-commandline (last visited 4/7/2024) (showing options for setting network speed properties).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.



`-netdelay` *`delay`* — Set network latency emulation to one of the following *`delay`* values in milliseconds:

- `gsm` - GSM/CSD (min 150, max 550).
- `hscsd` - HSCSD (min 80, max 400).
- `gprs` - GPRS (min 35, max 200).
- `edge` - EDGE/EGPRS (min 80, max 400).
- `umts` - UMTS/3G (min 35, max 200).
- `hsdpa` - HSDPA (min 0, max 0).
- `lte` - LTE (min 0, max 0).
- `evdo` - EVDO (min 0, max 0).
- none - No latency, the default (min 0, max 0).
- *`num`* - Specify exact latency.
- *`min`*`:`*`max`* - Specify individual minimum and maximum latencies.

For example:

```
emulator @Nexus_5X_API_23 -netdelay gsm
```

https://developer.android.com/studio/run/emulator-commandline (last visited 4/7/2024) (showing options for setting network latency properties).

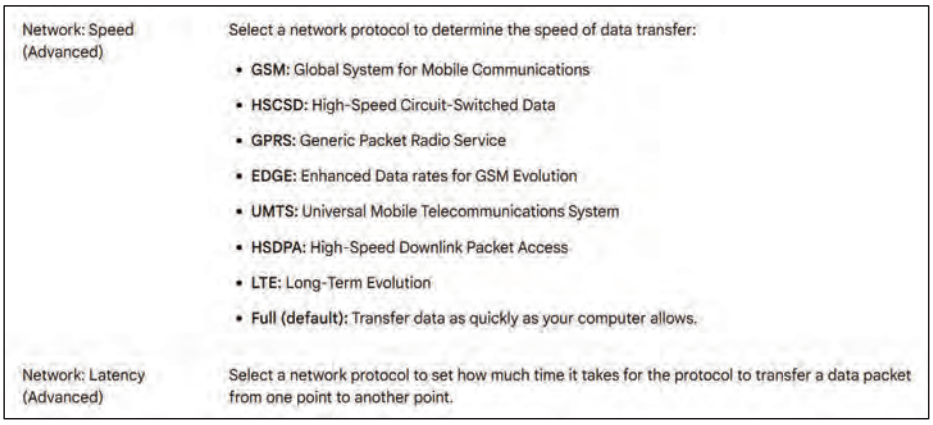
Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.

Claim 9

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.

The connection simulations of Android Studio are based on data of interaction with networks in non-simulated environments. The Network Speed and Latency settings offer a selection of specific protocols used in non-simulated network environments.



https://developer.android.com/studio/run/managing-avds (last visited 4/20/2024).

The network speed and latency options are based on data of interactions with networks in non-simulated environments at least because of their reliance on network standards developed to dictate the operation of real-world networks, such as LTE, HSDPA, UMTS, EDGE, GPRS, HSCSD, and GSM.

For instance, GSM corresponds to what is often referred to as the "2G" network standard established around 1991. Similarly, EDGE corresponds to Enhanced Data rates for GSM Evolution, an enhancement to GSM. And HSCSD corresponds to High Speed

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.

Circuit Switched Data, which is an enhancement to the data rate of circuit switched data in a GSM network. GPRS corresponds to a packet-oriented enhancement to 2G networks—General Packet Radio Service. UMTS is the Universal Mobile Telecommunications System, a new architecture that provided the basis for what is often referred to as the "3G" network standards. HSDPA, or High-Speed Downlink Packet Access, is an enhancement to the 3G network architecture to boost data capacity and improve download rates. Finally, LTE, or Long Term Evolution, represents the transition from 3G to what is typically referred to as "4G" network technology.

Each of these network standards defines the general operation of the network, and these definitions provide theoretical constraints on the networks' capacity for communication, including bandwidth, latency, and speed constraints. These constraints can be further impaired based on network conditions, including the presence of physical obstacles, electro-magnetic interference, and/or distance between the base station and a mobile station with which it is communicating.

The development and evolution of these standards relied on data of interactions with real-world implementations of such networks at least for testing and proof-of-concept. Thus, Android Studio's speed and latency constraints correspond to each identified standard, which are based on data of interaction of networks in non-simulated environments.

Many network operators have participated in (and continue to) the cellular standards detailed above. This participation includes both influencing the direction of the standards as well as testing the viability or early implementation of proposed standards.

> **a. The 3GPP Standards Development Process**
>
> Verizon is participating in and influencing the 5G standards setting process through the 3rd Generation Partner Project (3GPP), which has previously provided LTE, LTE-Advanced and LTE Advanced Pro for commercial cellular/mobile systems. There are seven organizational partners in 3GPP which work on the standards and also several peripheral organizations that reference or provide input to 3GPP standards (Figure 1).

https://www.verizon.com/business/resources/whitepapers/first-principles-for-securing-5g/ (last visited 5/17/2024); *see also id.* ("The standards development process, including work on security features, benefits from input from companies with real-world experience deploying new technology. It is common for companies like Verizon who are "first movers" to deploy service using new technology while the standards are still in development.").

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.

AT&T Teams Up with Global Technology Leaders for Faster 5G Deployment

AT&T* is working with several global technology leaders and operators to align on 5G. The efforts are in preparation for the release of the official 3GPP specifications which will form the basis of the global standards.

....

In 5G trials, AT&T is accelerating over-the-air interoperability testing based on standards developed under the 3GPP New Radio (NR) specifications. The trials are designed to easily evolve with future versions of the official 5G standards, a milestone 3GPP targets for 2018. Focusing on the NR standards helps ensure the technology will work correctly with any future 3GPP 5G NR updates.

https://about.att.com/story/faster_5g_deployment.html (last visited 5/17/2024).

Verizon has said that its goal is to collaborate with vendors on some early specifications and then contribute those to the 3GPP, the mobile industry standards body responsible for creating the 5G standard. Verizon insists that its early release of these 5G specifications won't create fragmentation in the industry.

....

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

9[A] The system of claim 2, wherein the one or more connection simulations are based on data of interaction with network operators in non-simulated environments.

> When Verizon released its 5G spec last July, the company said the guidelines were primarily for testing and validating 5G components that will help chipset vendors and others develop interoperable 5G gear and assist with pre-standard testing and fabrication.

https://www.sdxcentral.com/articles/news/verizon-att-spar-5g-standards-process/2016/09/ (last visited 5/17/2024).

When Android Studio was first released (2013), wireless networks had been widely deployed by network operators. On information and belief, Android Studio's speed and latency constraints are based on data of interaction with such networks in non-simulated environments operated by network operators.

These protocols are built for use in non-simulated environments, notably, operator networks. Therefore, Android Studio's Network Speed and Latency settings are based on data of interaction with network operators in non-simulated environments.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

12[A] The system of claim 1, wherein the software authoring interface is configured to allow a user to simulate an incoming sms message.

Claim 12

12[A] The system of claim 1, wherein the software authoring interface is configured to allow a user to simulate an incoming sms message.

Android Studio is configured to allow a user to simulate an incoming sms message.

> **Send a voice call or SMS to another emulator instance**
>
> The emulator automatically forwards simulated voice calls and SMS messages from one instance to another. To send a voice call or SMS, use the dialer app or SMS app, respectively, from one of the emulators.

https://developer.android.com/studio/run/emulator-networking (last visited 4/20/2024).

> To send an SMS message to another emulator instance:
>
> 1. Launch the SMS app, if available.
> 2. Specify the console port number of the target emulator instance as as the SMS address.
> 3. Enter the message text.
> 4. Send the message. The message is delivered to the target emulator instance.

https://developer.android.com/studio/run/emulator-networking (last visited 4/20/2024).

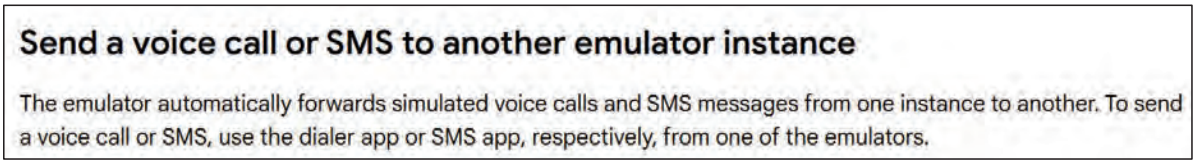
Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

13[A] The system of claim 1, wherein the software authoring interface is configured to allow a user to simulate an incoming phone call.

Claim 13

13[A] The system of claim 1, wherein the software authoring interface is configured to allow a user to simulate an incoming phone call.

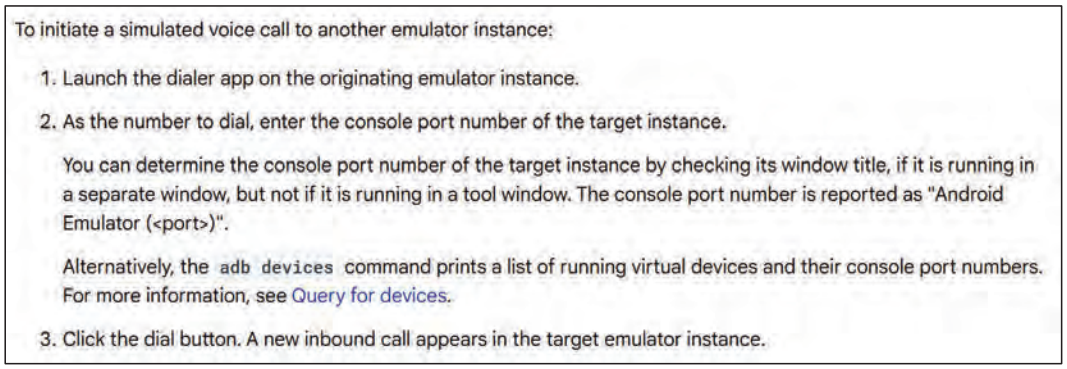
Android Studio is configured to allow a user to simulate an incoming phone call.



Send a voice call or SMS to another emulator instance

The emulator automatically forwards simulated voice calls and SMS messages from one instance to another. To send a voice call or SMS, use the dialer app or SMS app, respectively, from one of the emulators.

https://developer.android.com/studio/run/emulator-networking (last visited 4/20/2024).



To initiate a simulated voice call to another emulator instance:

1. Launch the dialer app on the originating emulator instance.
2. As the number to dial, enter the console port number of the target instance.

   You can determine the console port number of the target instance by checking its window title, if it is running in a separate window, but not if it is running in a tool window. The console port number is reported as "Android Emulator (<port>)".

   Alternatively, the `adb devices` command prints a list of running virtual devices and their console port numbers. For more information, see Query for devices.
3. Click the dial button. A new inbound call appears in the target emulator instance.

https://developer.android.com/studio/run/emulator-networking (last visited 4/20/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,

Claim 60

60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,

Defendant's mobile applications that it develops (including at least Chase Mobile) are configured to enable a user to modify a photo on the mobile device.

For example, upon information and belief, the check deposit feature of the Chase Mobile application is configured to enable a user to modify a photo on the mobile device. These modifications may include (but are not limited to): resizing, scaling, cropping, reformatting, filtering, and color adjustment.

As an example, the Chase Mobile application enables the user to capture a photo of a check and then modify the captured photo for review and submission to the bank for processing. An example demonstrating the modifications between the captured photo and the views of the check on the Quick Deposit View is shown in the images below.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

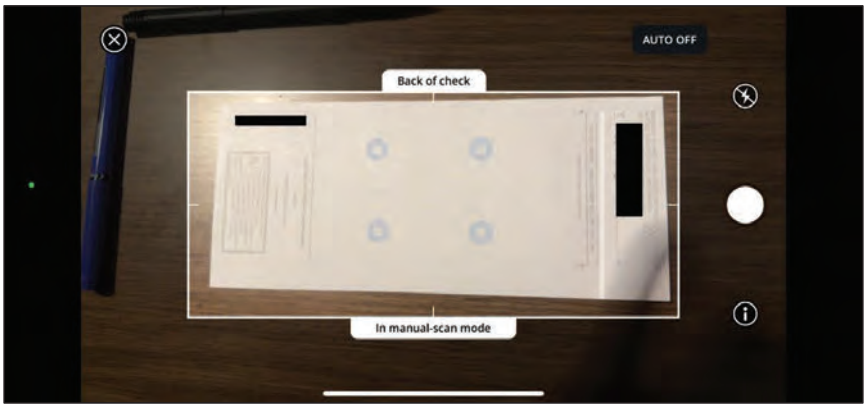
60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,

Screen Capture of Chase Mobile Application's Camera View, Front of Check - IOS (5/9/2024).[3]

[3] This screenshot is taken from an iOS app. The Chase Mobile application on Android does not permit screenshots while it is running. The iOS screenshot is representative of the operation of the Chase Mobile application on Android.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

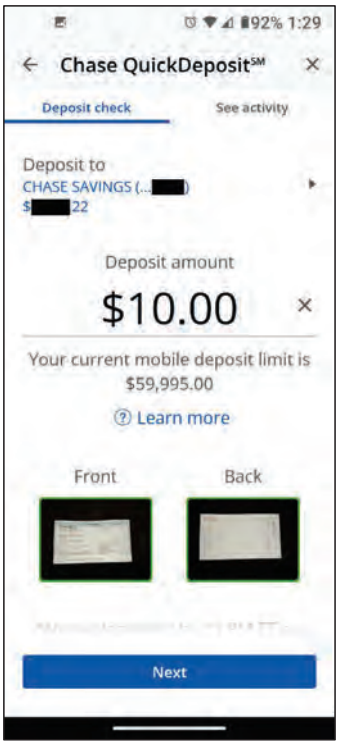
60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,



Screen Capture of Chase Mobile Application's Camera View, Back of Check - IOS (5/9/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,



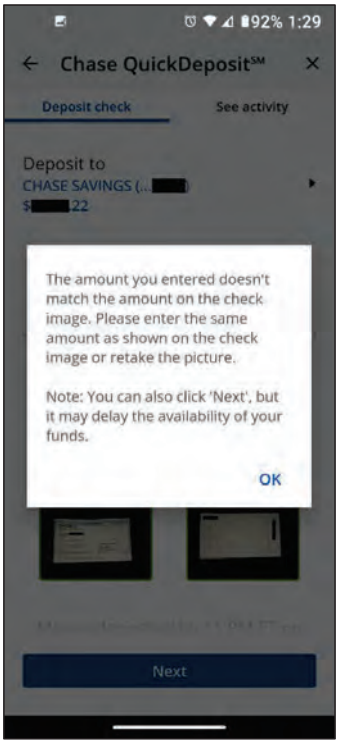
Screen Capture of Chase Mobile Application's Quick Deposit View, Front and Back Check Previews - Android (5/11/2024) (showing resized photos). As shown, the photos have been modified, including at least by being resized for review.

Upon information and belief, the Chase Mobile application also enables the user to modify the photo to perform functions such as optical character recognition on the photo in order to determine the amount shown on the check. The following image demonstrates

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,

that textual information (check amount) was read from the image in order to compare the manually entered deposit amount and compare it against the amount of the check, potentially resulting in modification of the photo.

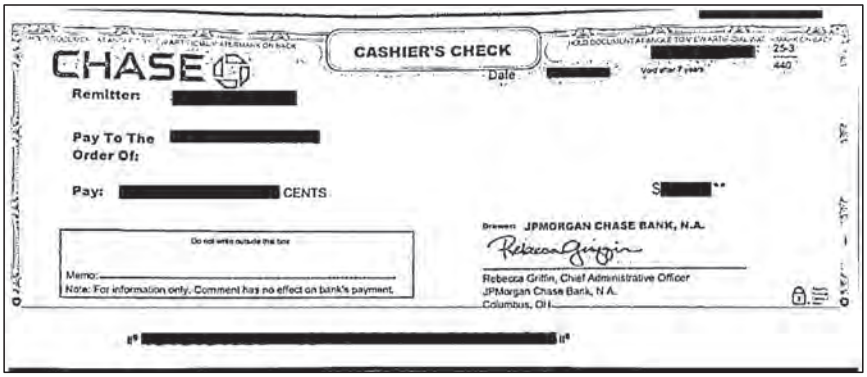


Screen Capture of Chase Mobile Application's Quick Deposit View with mismatch alert - Android (5/11/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,

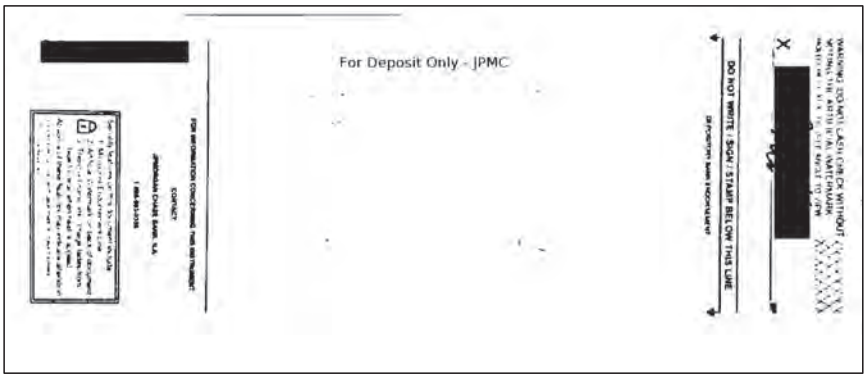
Further demonstrating modification of the check photos, deposited check photos available in the user's deposits history have been modified from the original, and all check photos appear to be cropped, resized, and color filtered.



Modified image of deposited check (front) from Chase server (5/16/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[A] A system comprising: an application configured to enable a user to modify a photo on the mobile device,



Modified image of deposited check (back) from Chase server (5/16/2024).

On information and belief, the Chase Mobile application prepares an Image Cash Letter (ICL) file format of the check for submission. *See* https://allmypapers.com/creating-icl-files-for-deposit/ (last visited 5/16/2024). Changing the format of the check is a modification of the photo, as is adding metadata (see claim 61 below, incorporated by reference). Chase previously deployed these capabilities in its IDD Mobile application in 2012. *See* https://jpmorganchaseco.gcs-web.com/news-releases/news-release-details/jp-morgan-launches-image-deposit-direct-mobile (last visited 5/16/2024). On information and belief, the Chase Mobile application modifies a captured check image to the ICL format.

Further evidence of infringement (including source code) is uniquely in the possession of Defendant, and Defendant has not yet made its source code available for inspection.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[B] wherein the application is developed using a software authoring platform configured to simultaneously visually emulate, via one or more profile display windows, a plurality of hardware characteristics indicative of performance of the mobile device when executing the application.

60[B] wherein the application is developed using a software authoring platform configured to simultaneously visually emulate, via one or more profile display windows, a plurality of hardware characteristics indicative of performance of the mobile device when executing the application.

As discussed above for limitations 1[A] and 1[B] (incorporated here by reference), Chase's applications (including at least Chase Mobile) are developed using a software authoring platform configured to simultaneously visually emulate, via one or more profile display windows, characteristics indicative of performance of the mobile device when executing the application.

Such characteristics include the "hardware profile properties" for an Android Virtual Device:

- Device Name
- Device Type
- Screen: Screen Size
- Screen: Screen Resolution
- Screen: Round
- Memory: RAM
- Input: Has Hardware Buttons (Back/Home/Menu)
- Input: Has Hardware Keyboard
- Input: Navigation Style
- Supported Device States
- Cameras
- Sensors: Accelerometer
- Sensors: Gyroscope
- Sensors: GPS

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[B] wherein the application is developed using a software authoring platform configured to simultaneously visually emulate, via one or more profile display windows, a plurality of hardware characteristics indicative of performance of the mobile device when executing the application.

- Sensors: Proximity Sensor
- Default Skin.

See https://developer.android.com/studio/run/managing-avds (last visited 4/6/2024) (listing and describing hardware profile properties). Additionally, an AVD has AVD properties that also control the manner in which the AVD performs during emulation/simulation. The AVD Properties include:

- AVD Name
- AVD ID (Advanced)
- Hardware Profile
- System Image
- Startup Orientation
- Camera (Advanced)
- Network: Speed (Advanced)
- Network: Latency (Advanced)
- Emulated Performance: Graphics
- Emulated Performance: Boot option (Advanced)
- Emulated Performance: Multi-Core CPU (Advanced)
- Memory and Storage: RAM (Advanced)
- Memory and Storage: VM Heap (Advanced)
- Memory and Storage: Internal Storage (Advanced)
- Memory and Storage: SD Card (Advanced)
- Device Frame: Enable Device Frame
- Custom Skin Definition (Advanced)
- Keyboard: Enable Keyboard Input (Advanced)

See https://developer.android.com/studio/run/managing-avds (last visited 4/6/2024) (listing and describing AVD properties). Each of the above-mentioned properties are emulated/simulated by Android Emulator when running the device profile specified for that

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

60[B] wherein the application is developed using a software authoring platform configured to simultaneously visually emulate, via one or more profile display windows, a plurality of hardware characteristics indicative of performance of the mobile device when executing the application.

particular AVD. The hardware profile properties and the AVD properties represent device characteristics, including hardware characteristics and network characteristics. When the Emulator runs a particular AVD, these characteristics are emulated/simulated and are indicative of the performance of the emulated/simulated device when running the application.

Android Studio includes numerous profile display windows showing, for example, CPU, Memory, Network, and Energy information about the performance of the mobile device when executing a mobile application as detailed in Claim 1[B].2.b above and incorporated fully herein by reference.

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

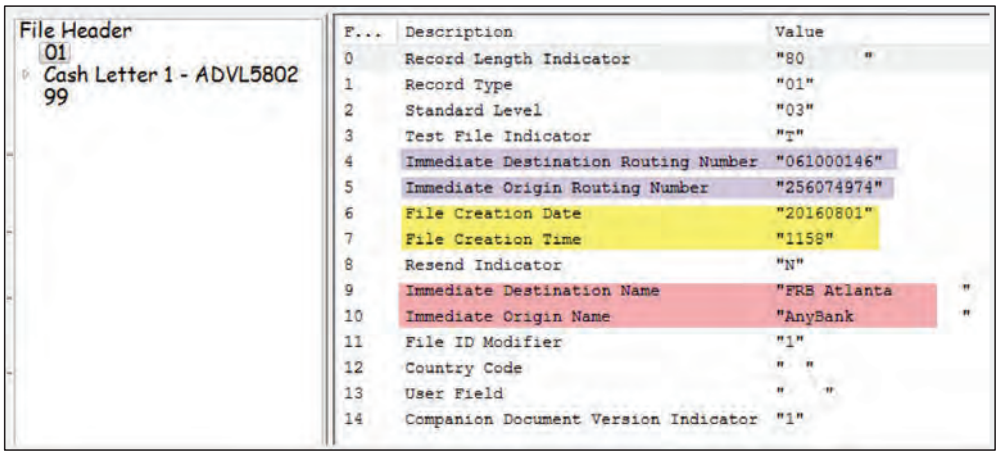
61[A] The system of claim 60, wherein the application is configured to allow an end user to add content to modify the photo.

Claim 61

61[A] The system of claim 60, wherein the application is configured to allow an end user to add content to modify the photo.

Defendant's mobile applications that it develops (including at least Chase Mobile) are configured to allow an end user to add content to modify the photo.

The ICL format includes metadata that includes content added to modify the captured photo. *See* 60[A] (detailing Chase support for the ICL image format). The ICL file includes metadata, as illustrated below:



https://allmypapers.com/creating-icl-files-for-deposit/ (last visited 5/16/2024).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

61[A] The system of claim 60, wherein the application is configured to allow an end user to add content to modify the photo.

Additionally, any OCR data added to the photo is content added to modify the captured photo. *See* 60[A] (detailing OCR of check image).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

62[A] The system of claim 61, wherein the content includes text.

Claim 62

62[A] The system of claim 61, wherein the content includes text.

Defendant's mobile applications that it develops (including at least Chase Mobile) are configured to allow an end user to add content to modify the photo wherein the content includes text.

See 61[A] (detailing metadata and OCR data content added to photos, both of which include text).

Infringement Chart for U.S. Patent No. 8,924,192 (Google's Android Studio Tools)

65[A] The system of claim 60, wherein the application is configured to allow an end user to distribute the modified photo through a server or other connection to the internet.

Claim 65

65[A] The system of claim 60, wherein the application is configured to allow an end user to distribute the modified photo through a server or other connection to the internet.

Defendant's mobile applications that it develops (including at least Chase Mobile) are configured to allow an end user to distribute the modified photo through a server or other connection to the internet.

Once the Chase Mobile application captures a check, the user can submit the check for processing. At this point, the modified photo is distributed to Chase servers through a wireless connection to the internet. *See* 60[A] (detailing check capture and submission).